Sécurité RDP : Intecteption d'authentification sur NLA avec CredSSPy

Geoffrey Bertoli – @yofbalibump (Synacktiv)
Thomas Bourguenolle (EY)

SYNACKTIV DIGITAL SECURITY

EY

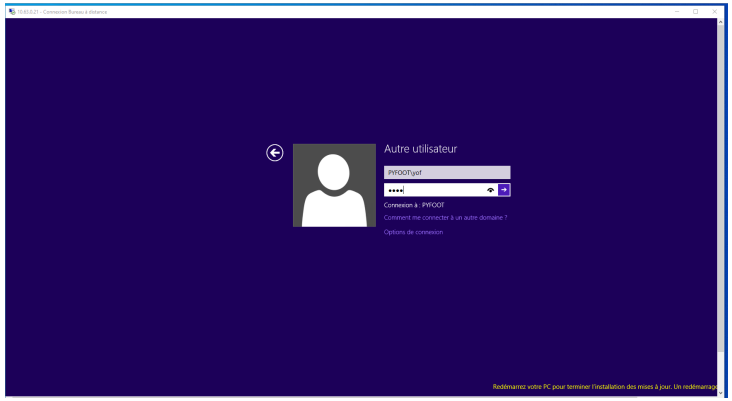
Table des matières

Mode d'authentification pour RDP

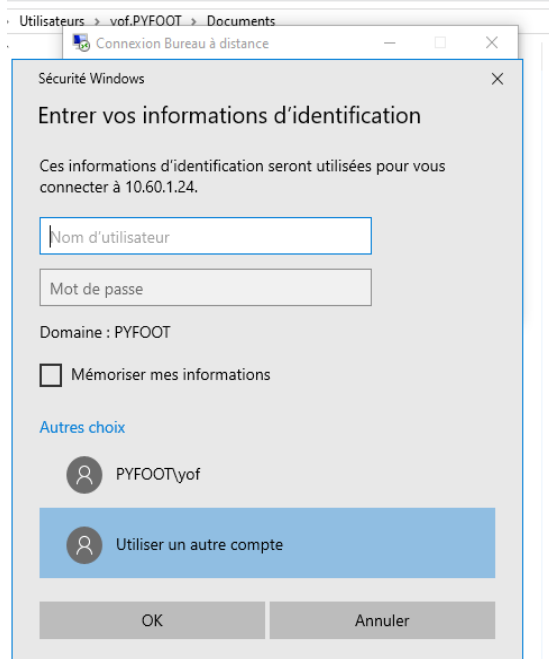
- Les serveurs RDP supportent actuellement 3 modes d'authentification distincts pour l'établissement d'une connexion :
  - **RDP Standard Security** : Les échanges entre le client et le serveur sont chiffrés à l'aide d'une clé symétrique. Cette clé est négociée à partir d'un échange RSA établi avec une paire de clés fixe spécifiée dans la documentation Microsoft.
  - **RDP Enhanced sécurity TLS** (Depuis RDP 5.2) : Les échanges entre le client et le serveur sont chiffrés à l'aide d'une clé symétrique négociée dans un canal TLS.
  - **RDP Enhanced security NLA** (Depuis RDP 6.0) Mode d'authentification par défaut depuis Windows Server 2012R2. Basé sur CredSSP, les échanges entre le client et le serveur sont chiffrés à l'aide d'une clé symétrique négociée dans un canal TLS après « authentification mutuelle » des deux acteurs (via Kerberos ou NTLMSSP)

Mode d'authentification pour RDP

- Authentication pré-NLA



- Authentification avec NLA



SYNACKTIV DIGITAL SECURITY

EY

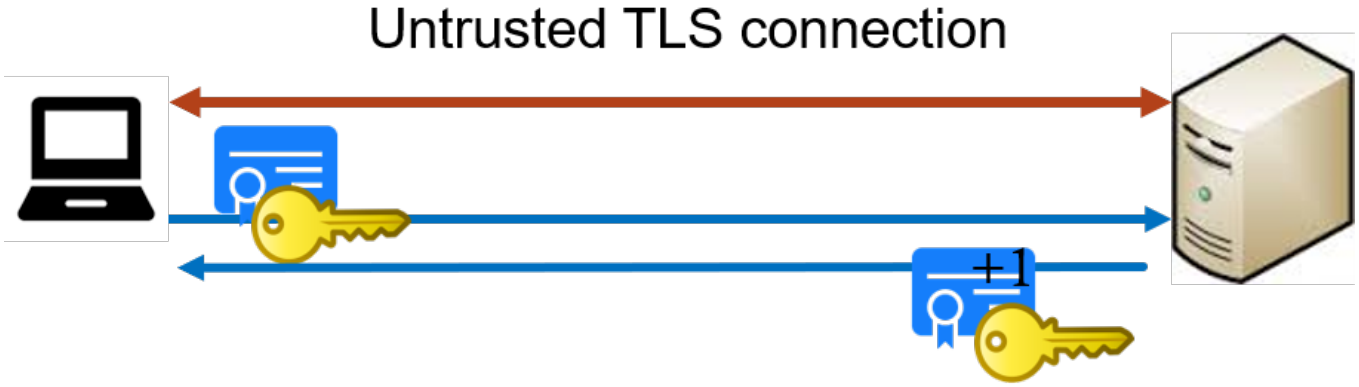
- Plusieurs outils permettent actuellement la mise en place d'attaques de type Man-In-The-Middle sur RDP
- SETH, PyRDP ou encore rdpy nécessitent de forcer le client à utiliser un mode d'authentification moins sécurisé
- Un article présenté à SSTIC 2012 [1], indique cependant la possibilité d'une telle attaque dans le cas ou une machine du domaine serait compromise

1. Sécurité de RDP : `https://www.sstic.org/2012/presentation/securite_rdp/`

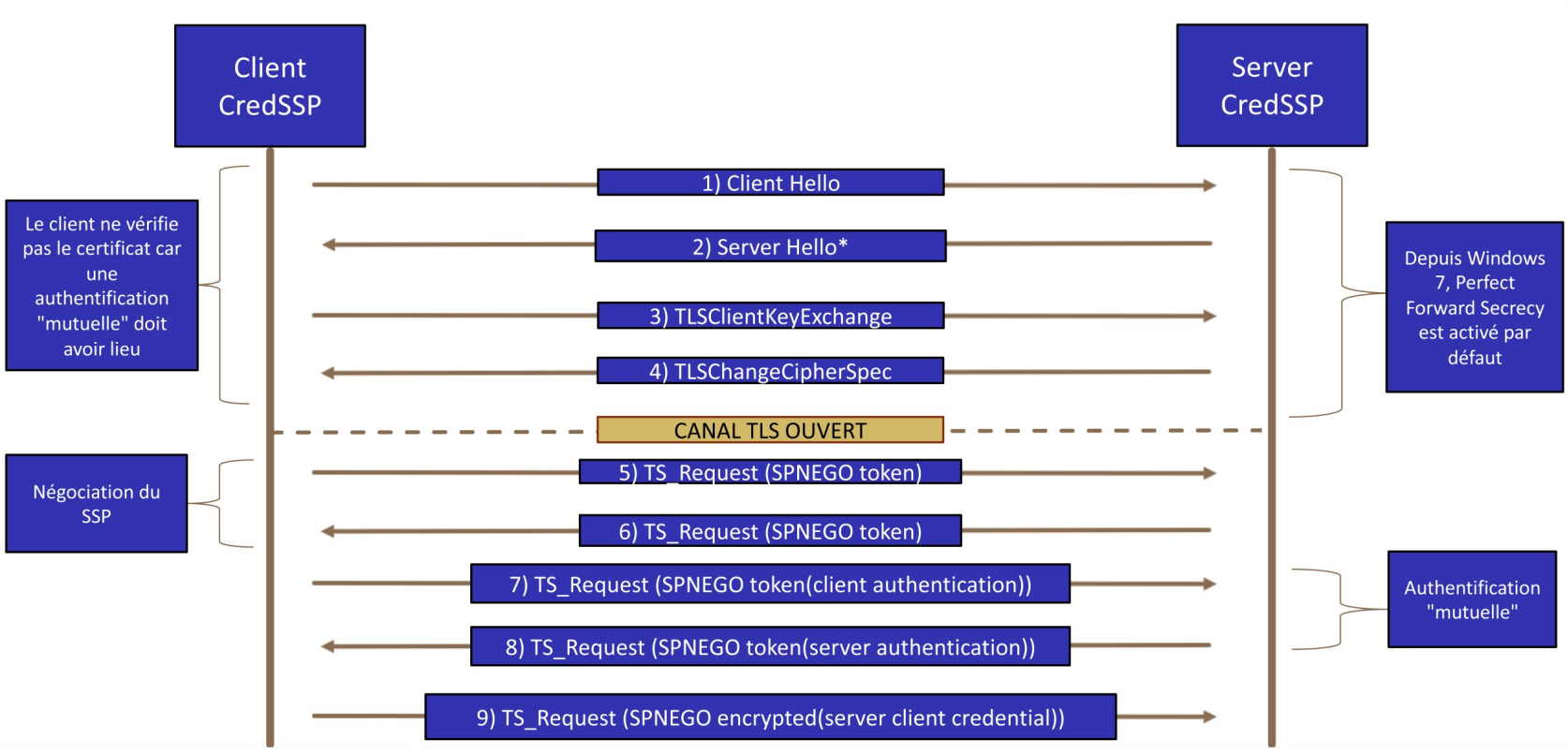
Table des matières

Le protocole Credential Security Support Provider (CredSSP) permet à une application de transmettre de manière sécurisé des authentifiants à une application cible

Le protocole utilise d'abord un canal TLS non approuvé. Ensuite à l'aide du protocole SPNEGO une authentification « mutuelle » du serveur et du client est effectuée Les authentifiants (ici ceux de l'utilisateur de la session RDP) sont ensuite envoyés à l'intérieur de ce canal TLS



CredSSP - Description du protocole



SYNACKTIV DIGITAL SECURITY

EY

CredSSP - TS_Request

- La TSRequest est la structure principale d'un paquet CredSSP. Elle encapsule le jeton SPNEGO ainsi que les potentiels messages Kerberos ou NTLM échangés entre le serveur et le client
- TSRequest structure :

```
TSRequest ::= SEQUENCE {

        version       [0] INTEGER,

        negoToken     [1] NegoData OPTIONNAL,

        authInfo
[2] OCTET STRING OPTIONNAL,

        pubKeyAuth
[3] OCTET STRING OPTIONNAL,

        errorCode
[4] OCTET STRING OPTIONNAL,

        clientNonce [5] OCTET STRING OPTIONNAL

        }
```

- La présentation traitera la version 6 de CredSSP
- Depuis la vulnérabilité CVE-2018-0886, le client Microsoft refuse les connexions à un serveur utilisant la version 5 ou inférieure (par défaut)

CredSSP - Description du protocol

- Pendant l'étape 5 et 6, le serveur et le client négocient quel SSP utiliser. Il est possible d'utiliser NLTMSSP ou Kerberos

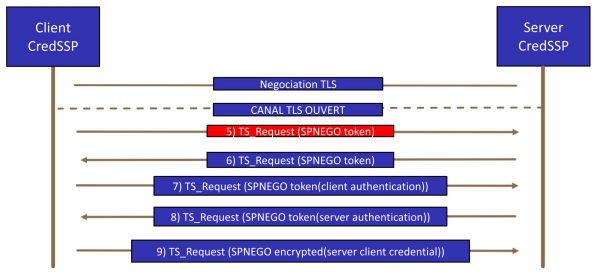
- NTLMSSP sera choisi si :
  - Le client n'appartient pas à un domaine **OU**
  - Le client utilise l'adresse IP pour se connecter au serveur

- Kerberos sera choisi si :
  - Le client et le domaine appartiennent au même domaine **ET**
  - Le client utilise le FQDN du serveur pour la connexion

Table des matières

CredSSP - NTLMSSP Etape 5

- Le client envoie un message indiquant qu'il souhaite utiliser NTLM comme SSP




```
TSRequest {
        Version:      6,
        negoToken:
NegoData-> NTLMSS NEGOTIATE\_MESSAGE
        }
```

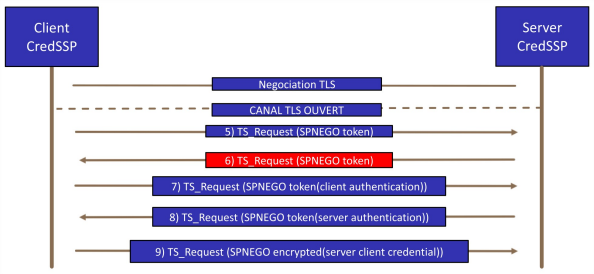
- Le NTLMSSP NEGOTIATE_MESSAGE contient les informations sur la version de NTLMSSP supportée ainsi que les flags à utiliser

SYNACKTIV DIGITAL SECURITY

EY

CredSSP - NTLMSSP Etape 6

- Le serveur renvoie alors un challenge NTLM

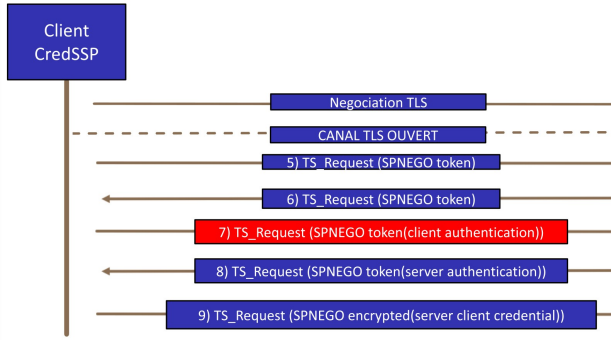



```
TSRequest {
        Version:      6,
        negoToken:
NegoData-> NTLMSSP CHALLENGE\_MESSAGE
        }
```

- NTLMSSP CHALLENGE_MESSAGE est le challenge NTLM contenant :
  - NTLMChallenge
  - Server name
  - Domain name

CredSSP - NTLMSSP Etape 7

- Le client répond au challenge et demande une vérification de l'intégrité de l'échange TLS




```
TSRequest {
        Version:      6,
        negoToken:
NegoData-> NTLMSSP AUTHENTICATE\_MESSAGE,
        pubKeyAuth: Signature + encryptedpubKey,
        clientNonce: Octet String
        }
```

- NTLMSSP AUTHENTICATE_MESSAGE est la réponse au challenge contenant :
  - NTLMv2 response
  - Username
  - Domain name
  - EncryptedSessionKey = $\underset{NTLMKey}{RC4}$(RandomSessionKey) -

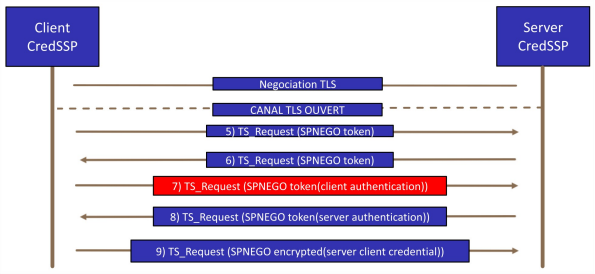
    NTLMKey : HMACMD5 derivé du NTLM Challenge et creds

SYNACKTIV DIGITAL SECURITY

EY

CredSSP - NTLMSSP Etape 7

- Le client répond au challenge et demande une vérification de l'intégrité de l'échange TLS




```
TSRequest {
        Version:      6,
        negoToken:
NegoData-> NTLMSSP AUTHENTICATE\_MESSAGE,
        pubKeyAuth: Signature + encryptedpubKey,
        clientNonce: Octet String
        }
```

- pubKeyAuth contient :
  - $RC4_{RandomSessionKey}$(SHA256('CredSSP Client-To-Server Binding Hash' + Nonce + PublicKey))

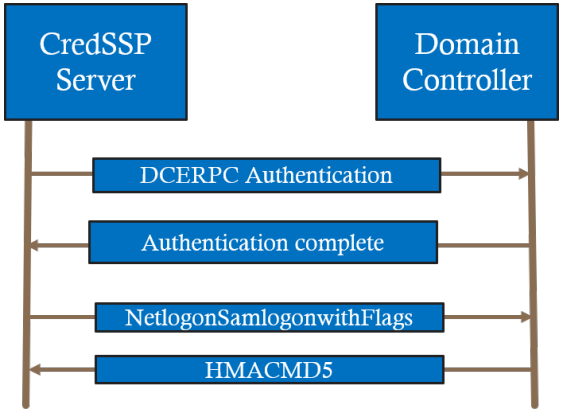
Le serveur peut alors vérifier l'intégrité de la connexion TLS en vérifiant le champs pubKeyAuth. Pour cela il a besoin de la valeur de RandomSessionKey (envoyée par le client mais chiffrée à l'aide de la NTLMKey)

SYNACKTIV DIGITAL SECURITY

EY

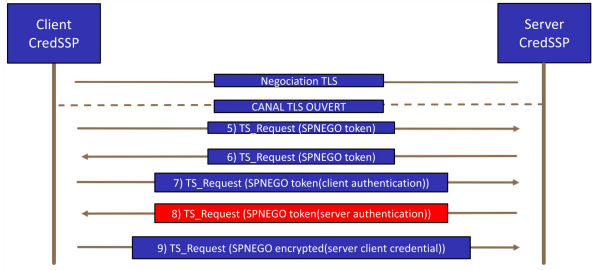
CredSSP - NTLMSSP Etape 7 (vérification serveur)

- Si le compte appartient au domaine :
  - Le serveur s'authentifie auprès du DC via DCE-RPC
  - Le serveur transmet la réponse NTLMv2 auprès du DC (NetLogonSamLogonwithFlags request)
  - Le DC authentifie le client et envoie au serveur la NTLMKey
  - Le serveur déchiffre alors encryptedSessionKey et récupère la RandomSessionKey
  - Le serveur déchiffre PubKeyAuth et vérifie l'intégrité du canal TLS



CredSSP - NTLMSSP Etape 8

```
TSRequest {
        Version:      6,
        pubKeyAuth: Signature + encryptedCertificate,
        }
```



- pubKeyAuth contient :
  - $RC4_{RandomSessionKey}$(SHA256('CredSSP Server-To-Client Binding Hash' + Nonce + PublicKey))
  - Les différences sur les versions de CredSSP sont notables sur la manière de chiffrer ces informations

- **Conclusion :** « L'authentification mutuelle » permet seulement de vérifier que le serveur peut récupérer la NTLMKey i.e. qu'il est en mesure de s'authentifier auprès du DC

SYNACKTIV DIGITAL SECURITY

EY

CredSSP - NTLMSSP Step 8

Le client va-t'il envoyer ses authentifiants maintenant ?

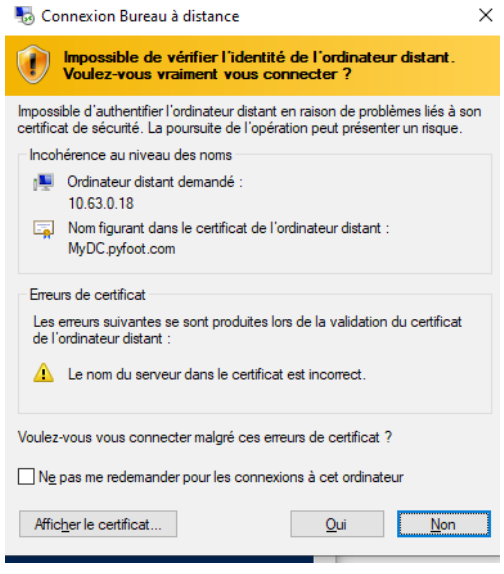
- Non, cette vulnérabilité a été patchée dans RDP 7.0

L'ensemble du process va recommencer une deuxième fois et montrer le certificat à l'utilisateur

Cependant l'attaquant en position de singe intercepteur aura déjà obtenu une réponse NTLMv2 valide (en vue d'un cassage ou d'un relai)

CredSSP - NTLMSSP Etape 9

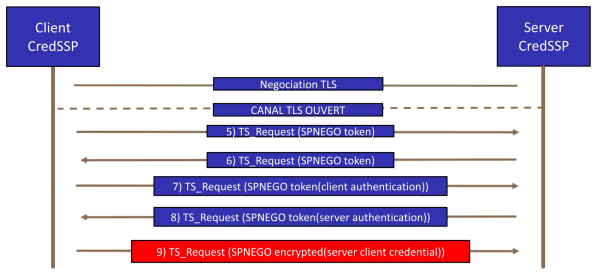
Dans le cas ou le client accepte le certificat, l'échange recommence et va jusque l'étape 9



N.B. : L'utilisation de l'adresse IP par le client pour la connexion auprès du serveur, générera dans le cas général une erreur de certificat

CredSSP - NTLMSSP Etape 9

```
TSRequest {
        Version:      6,
        authInfo: Signature + cryptedCreds,
        }
```



- AuthInfo contient :
  - $RC4_{RandomSessionKey}$(Credentials Structure)
- Credentials Structure contient le mode d'authentification (password, smartcard,...) et les authentifiants

CredSSP - Demo

DEMO

Table des matières

Quelles sont les différences ?

- Le NegoData contient maintenant un TGS valide à destination du serveur contenant la randomSessionKey
- De fait il n'est plus possible d'utiliser n'importe quel compte machine pour réaliser l'attaque
- Cependant, il est possible de partiellement downgrader cette connexion

Lors de la négociation du SSP, la TSRequest du client est de cette forme :

```
TSRequest {
        Version:     6,
        NegoToken: Kerberos,
        }
```

Même si le client supporte NTLMSSP, il n'est pas proposé par le client

Cependant il est possible de forcer le client à utiliser NTLMSSP en envoyant un paquet SPNEGO_RENOGOCIATE

```
TSRequest {
        Version:      6,
        NegoToken:  \xa1\x...NTLMSSP\x00\x00...\x0f,
        }
```

À réception de ce message, le client accepte maintenant d'utiliser NTLMSSP mais... le protocole CREDSSP est maintenant complétement « cassé »

Pour des raisons inconnues, le message NTLM NEGOTIATE_MESSAGE est maintenant dans le champ AuthInfo dans une TSRequest encapsulée dans le champ « NegoData »

```
TSRequest {
        Version:      6,
        NegoToken: TSRequest{
                Version: 1
                AuthInfo : NTLMSSP NEGOCIATE\_MESSAGE
                }
        }
```

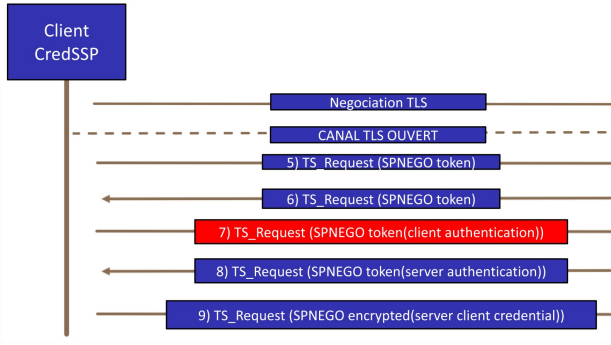
Il suffit maintenant de répondre avec un challenge NTLMv2 et d'utiliser la même structure pour l'envoi des informations :

```
TSRequest {
        Version:      6,
        NegoToken: TSRequest{
                Version: 1
                AuthInfo : NTLMSSP CHALLENGE\_MESSAGE
                }
        }
```

CredSSP - Kerberos Step 7

Le client répondra avec sa réponse NTLMv2 suivant la même structure

```
TSRequest {
        Version:      6,
        NegoToken: TSRequest{
                Version: 1
                AuthInfo : NTLMSSP AUTHENTICATE\_MESSAGI
                },
        pubKeyAuth : NULL
        ClientNonce : NULL
        }
```



Cependant il n'envoie aucune information Cryptographique (RandomSessionKey, etc.), il n'est donc pas possible de répondre de manière valide auprès du client

La connexion va alors échouer mais l'attaquant a obtenu une réponse NTLMv2 valide sans générer d'alerte sur le certificat de la connexion

CredSSP - Downgrading Kerberos Auth

DEMO

- Protected Users ajoute des protections sur la gestion de l'authentification :
  - Empêche de se loguer via NTLM
  - Empêche le cache d'authentifiants
  - Empêche l'utilisation de RC4 (NTLM) pour la pré-authentification Kerberos
- À titre d'exemple, lorsqu'un Protected User utilise une adresse IP pour se connecter à un partage réseau, aucune requête NTLMSSP n'est envoyée

Il est tout de même possible de récupérer une réponse NTLMv2 d'un utilisateur Protected User, il ne sera pas contre pas possible de faire du relai avec

- L'outil ne fonctionne actuellement que contre le client officiel Microsoft :
  - Le client Mac Microsoft et FreeRDP vérifie le certificat à la première connexion
  - Le client Mac Microsoft et FreeRDP ne comprennent pas le paquet permettant le downgrade Kerberos
- L'outil est toujours en Beta, des évolutions pourrait être :
  - Ajouter les fonctionnalités d'authentification NLA dans rdpy ou PyRDP afin de profiter des fonctions d'enregistrements de sessions ou de capture des frappes claviers
  - Implémenter le RelayNTLMSSP

Questions

SYNACKTIV DIGITAL SECURITY

EY